

**UNITED STATES PATENT APPLICATION**

of

**Shawn D. Bracewell**

**Troy Mayhan**

**Russell L. Simpson, Jr.**

**Puhazholi Vetrivel**

and

**Ward Beattie**

for

**TEMPLATE ARCHITECTURE AND RENDERING ENGINE  
FOR WEB BROWSER ACCESS TO DATABASES**

**WORKMAN, NYDEGGER & SEELEY**  
A PROFESSIONAL CORPORATION  
ATTORNEYS AT LAW  
1000 EAGLE GATE TOWER  
60 EAST SOUTH TEMPLE  
SALT LAKE CITY, UTAH 84111

U.S. PATENT & TRADEMARK OFFICE

## **BACKGROUND OF THE INVENTION**

### **1. Cross Reference to Related Applications**

This application claims priority from United States Provisional Application Serial number 60/237,259, entitled TEMPLATE ARCHITECTURE AND RENDERING ENGINE FOR WEB BROWSER BASED ACCESS TO EXCHANGE EMAIL, and filed October 2, 2000, which provisional application is incorporated herein in its entirety.

### **2. The Field of the Invention**

The present invention relates to displaying data of varied formats using a web browser. More specifically, the present invention relates to displaying data on a Web browser, the data not being natively compatible with display on the web browser, and where the display is facilitated using a template.

### **3. Background and Related Art**

The popularity of the Internet has profoundly improved the way people communicate by allowing users quick and easy access to information. By accessing the World Wide Web and electronic mail through computers and other devices, people now stay in touch with each other around the globe, and can access information on a virtually limitless variety of subjects.

With the ever-increasing use of the World Wide Web, there is a need to securely and efficiently display data stored in a variety of differing formats using a web interface such as a web browser. This includes simultaneously displaying data where the data is stored in formats that are not natively compatible with display using a web browser.

1 Current methods for displaying non-web data in a web format employ generalized  
2 data binding techniques. While such products are capable of displaying a variety of data  
3 formats, their generality leads to inefficiency when accessing certain types of data, which  
4 places an unneeded strain on the resources of such systems that use general-purpose data  
5 collection products. Additionally, adding support to display data in different languages  
6 requires extensive recoding, which is an inefficient use of programming resources. Also,  
7 due to the operation of such products, the object modules associated with these methods  
8 are generally accessible during operation, which decreases the security of systems that  
9 employ such methods.

10 Accordingly, what are desired are systems, methods, computer program products,  
11 and data structures that facilitate more secure Web access to data that is not natively  
12 compatible with a Web browser, and where changes to the layout (such as language  
13 changes or layout changes that depend on the type of Web browser) on the Web browser  
14 are more easily realized without extensive recoding.

## SUMMARY OF THE INVENTION

The present invention relates to systems, methods, computer program products, and data structures for efficiently rendering data that is not natively compatible for viewing on web browsers. The rendering may be performed across a variety of different web browsers and languages with minimum recoding, and with reduced security risk.

The present invention uses "templates." By using templates, it is possible to view a particular set of data on a Web browser where normally the set of data would not be viewable. A template is used to define a web page including how data not normally viewable on a Web browser should be processed to make it viewable. A template may include references to data stored in a database and functions that process the referenced data to produce displayable data.

Templates are created by including custom tags, which are references to other subroutines, and processor directives in HTML documents. Processor directives may instruct the processor to perform certain operations when generating a template, such as generating a template for different types of Web browsers. After a template is generated it may be associated with any number of language support files. When additional language support files are added, the template may be associated with these added files without having to re-generate the template. In a Microsoft® Windows® environment this may be as simple as copying a new file onto an associated computer system.

Using templates makes accessing data more efficient because templates may be custom generated for use with specific data. Since adding functionality for a new language is often as simply as copying a new file onto a computer system, templates also provide an efficient way to display data in a variety of languages. Furthermore, since templates are compiled, they are essentially unreadable, which makes it difficult for

1 unauthorized users to determine what data the template is accessing and modify the data in  
2 any way. Additionally, the object code associated with templates is stored in a location  
3 different from associated HTML code. Thus, even if an unauthorized person were to gain  
4 access to the HTML code, all the person would see is the customized tag that is included in  
5 the HTML document. There would be no way for the unauthorized person to modify any  
6 code associated with the functionality of the customized tag.

7       Templates may typically be used in a network environment such as a LAN, WAN  
8 or on the Internet. These environments include multiple interconnected computer systems,  
9 which may typically need to access stored data. In an Internet environment, it may also be  
10 the case that a number of computer systems are employing Web browsers to access data  
11 stored on network servers or data servers, across the World Wide Web. A user may have  
12 access to a computer system including a Web browser, but the computer system does not  
13 include programs needed to display data from proprietary e-mail or other databases and  
14 this data is not natively displayable using a Web browser.

15       In accordance with the present invention, a user may make a request to a network  
16 server requesting such proprietary data. Included in the request is information that allows  
17 the network server to identify what browser type and language the requested content  
18 should be returned in. For example, if the request is an HTTP request, the Accept-  
19 Language HTTP header may be used to determine the language, and the User-Agent HTTP  
20 header may be used to determine the browser type.

21       When the network server receives the request, it determines which template is  
22 associated with displaying the requested content for the Web browser type. A template  
23 may include displayable content such as HTML code, which may display fixed items such  
24 as the "TO" or "FROM" fields when viewing e-mail data. The template may also include

1 non-displayable content, such as functions that retrieve and process other non-displayable  
2 data from a data server.

3 After the correct template is identified, the network server may access non-  
4 displayable data from the data server. Such non-displayable data may be an e-mail  
5 message that is formatted for use using a specific e-mail client that is not compatible with a  
6 Web browser or a word processing document which would normally be viewed using a  
7 word processing program such as MICROSOFT® Word.

8 Once the data is retrieved, the network server constructs the requested displayable  
9 content. This may include processing the displayable content as specified in the identified  
10 template. The processing may include using the non-displayable data as input for one or  
11 more functions so that it may be converted into a format that is viewable using a Web  
12 browser. This may also involve intermixing already displayable portions of the template  
13 with the processed, formerly non-displayable, content to create the requested displayable  
14 content. Once the requested content is in a format for Web browser viewing, the network  
15 server sends the displayable content to the Web browser.

16 Additional features and advantages of the invention will be set forth in the  
17 description which follows, and in part will be obvious from the description, or may be  
18 learned by the practice of the invention. The features and advantages of the invention may  
19 be realized and obtained by means of the instruments and combinations particularly  
20 pointed out in the appended claims. These and other features of the present invention will  
21 become more fully apparent from the following description and appended claims, or may  
22 be learned by the practice of the invention as set forth hereinafter.

## **BRIEF DESCRIPTION OF THE DRAWINGS**

In order to describe the manner in which the above recited and other advantages and features of the invention can be obtained, a more particular description of the invention briefly described above will be rendered by reference to specific embodiments thereof, which is illustrated, in the appended drawings. Understanding that these drawings depict only typical embodiments of the invention and are not therefore to be considered to be limiting of its scope, the invention will be described and explained with additional specificity and detail through the use of the accompanying drawings in which:

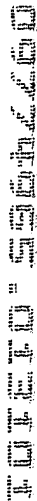
Figure 1 illustrates an exemplary system that provides a suitable operating environment for the present invention;

Figure 2 illustrates some of the functional components present in a network system where data from a data server may be rendered to a network device;

Figure 3 is a flow diagram illustrating a method whereby data from a data server is rendered on a network device;

Figure 4 is an example of HTML code including customized tags that illustrates how a template may be constructed; and

Figure 5 is an example of a data structure that illustrates a representation of a data layout for a template.



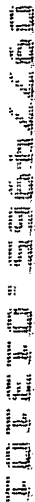
## DETAILED DESCRIPTION OF THE INVENTION

The present invention extends to methods, systems, computer program products, and data structures data for rendering content in order to make it displayable on a Web browser. Templates are used to access and format data that would otherwise not be displayable on a Web browser. The template architecture reduces the coding time needed to create templates for use with different types of Web browsers and for use with a variety of different languages. Templates are compiled from native HTML code as well as customized tags, these customized tags are processed to do the actually data gathering.

When a network server receives a request for data, the network server determines what template to use based on the browser type, language, action requested, and object state. At run time, a rendering engine processes the customized tags to access and format data that will be sent to a Web browser.

The embodiments of the present invention may comprise a special purpose or general-purpose computer including various computer hardware components, as discussed in greater detail below. Embodiments within the scope of the present invention also include computer-readable media for carrying or having computer-executable instructions or data structures stored thereon. Such computer-readable media can be any available media, which can be accessed by a general purpose or special purpose computer. By way of example, and not limitation, such computer-readable media can comprise physical storage media such as RAM, ROM, EEPROM, CD-ROM or other optical disk storage, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to carry or store desired program code means in the form of computer-executable instructions or data structures and which can be accessed by a general purpose or special purpose computer.





1 When information is transferred or provided over a network or another  
2 communications connection (either hardwired, wireless, or a combination of hardwired or  
3 wireless) to a computer, the computer properly views the connection as a computer-  
4 readable medium. Thus, any such connection is properly termed a computer-readable  
5 medium. Combinations of the above should also be included within the scope of  
6 computer-readable media. Computer-executable instructions comprise, for example,  
7 instructions and data which cause a general purpose computer, special purpose computer,  
8 or special purpose processing device to perform a certain function or group of functions.

9 Figure 1 and the following discussion are intended to provide a brief, general  
10 description of a suitable computing environment in which the invention may be  
11 implemented. Although not required, the invention will be described in the general context  
12 of computer-executable instructions, such as program modules, being executed by  
13 computers in network environments. Generally, program modules include routines,  
14 programs, objects, components, data structures, etc. that perform particular tasks or  
15 implement particular abstract data types. The sequence of instructions implemented in a  
16 particular data structure or program module represents examples of corresponding acts for  
17 implementing the functions or steps described herein.

18 Those skilled in the art will appreciate that the invention may be practiced in  
19 network computing environments with many types of computer system configurations,  
20 including personal computers, hand-held devices, multi-processor systems,  
21 microprocessor-based or programmable consumer electronics, network PCs,  
22 minicomputers, mainframe computers, and the like. The invention may also be practiced  
23 in distributed computing environments where tasks are performed by local and remote  
24 processing devices that are linked (either by hardwired links, wireless links, or by a

1 combination of hardwired or wireless links) through a communications network. In a  
2 distributed computing environment, program modules may be located in both local and  
3 remote memory storage devices.

4 With reference to Figure 1, an exemplary system for implementing the invention  
5 includes a general-purpose computing device in the form of a conventional computer 120,  
6 including a processing unit 121, a system memory 122, and a system bus 123 that couples  
7 various system components including the system memory 122 to the processing unit 121.  
8 The system bus 123 may be any of several types of bus structures including a memory bus  
9 or memory controller, a peripheral bus, and a local bus using any of a variety of bus  
10 architectures. The system memory includes read only memory (ROM) 124 and random  
11 access memory (RAM) 125. A basic input/output system (BIOS) 126, containing the basic  
12 routines that help transfer information between elements within the computer 120, such as  
13 during start-up, may be stored in ROM 124.

14 The computer 120 may also include a magnetic hard disk drive 127 for reading  
15 from and writing to a magnetic hard disk 139, a magnetic disk drive 128 for reading from  
16 or writing to a removable magnetic disk 129, and an optical disk drive 130 for reading  
17 from or writing to removable optical disk 131 such as a CD-ROM or other optical media.  
18 The magnetic hard disk drive 127, magnetic disk drive 128, and optical disk drive 130 are  
19 connected to the system bus 123 by a hard disk drive interface 132, a magnetic disk drive-  
20 interface 133, and an optical drive interface 134, respectively. The drives and their  
21 associated computer-readable media provide nonvolatile storage of computer-executable  
22 instructions, data structures, program modules and other data for the computer 120.  
23 Although the exemplary environment described herein employs a magnetic hard disk 139,  
24 a removable magnetic disk 129 and a removable optical disk 131, other types of computer

1 readable media for storing data can be used, including magnetic cassettes, flash memory  
2 cards, digital versatile disks, Bernoulli cartridges, RAMs, ROMs, and the like.

3       Program code means comprising one or more program modules may be stored on  
4 the hard disk 139, magnetic disk 129, optical disk 131, ROM 124 or RAM 125, including  
5 an operating system 135, one or more application programs 136, other program modules  
6 137, and program data 138. A user may enter commands and information into the  
7 computer 120 through keyboard 140, pointing device 142, or other input devices (not  
8 shown), such as a microphone, joy stick, game pad, satellite dish, scanner, or the like.  
9 These and other input devices are often connected to the processing unit 121 through a  
10 serial port interface 46 coupled to system bus 123. Alternatively, the input devices may be  
11 connected by other interfaces, such as a parallel port, a game port or a universal serial bus  
12 (USB). A monitor 147 or another display device is also connected to system bus 123 via  
13 an interface, such as video adapter 148. In addition to the monitor, personal computers  
14 typically include other peripheral output devices (not shown), such as speakers and  
15 printers.

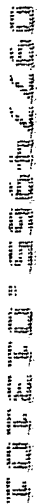
16       The computer 120 may operate in a networked environment using logical  
17 connections to one or more remote computers, such as remote computers 149a and 149b.  
18 Remote computers 149a and 149b may each be another personal computer, a server, a  
19 router, a network PC, a peer device or other common network node, and typically include  
20 many or all of the elements described above relative to the computer 120, although only  
21 memory storage devices 150a and 150b and their associated application programs 136a and  
22 136b have been illustrated in Figure 1. The logical connections depicted in Figure 1  
23 include a local area network (LAN) 151 and a wide area network (WAN) 152 that are  
24 presented here by way of example and not limitation. Such networking environments are

1 commonplace in office-wide or enterprise-wide computer networks, intranets and the  
2 Internet.

3 When used in a LAN networking environment, the computer 120 is connected to  
4 the local network 151 through a network interface or adapter 153. When used in a WAN  
5 networking environment, the computer 120 may include a modem 154, a wireless link, or  
6 other means for establishing communications over the wide area network 152, such as the  
7 Internet. The modem 154, which may be internal or external, is connected to the system  
8 bus 123 via the serial port interface 146. In a networked environment, program modules  
9 depicted relative to the computer 120, or portions thereof, may be stored in the remote  
10 memory storage device. It will be appreciated that the network connections shown are  
11 exemplary and other means of establishing communications over wide area network 152  
12 may be used.

13 In this description and in the following claims, the term "computer" should be  
14 interpreted broadly to encompass a general purpose or special purpose computer or any  
15 other computing device including, but not limited to, various computer hardware  
16 components, such as those illustrated in Figure 1. A "computer system" includes a group  
17 of one or more computers that interact cooperatively to perform one or more functions. A  
18 "network system" includes a plurality of interconnected computer systems, as well as other  
19 devices that may be connected to computer systems. A "client system" is a computer  
20 system that uses the services of another computer system. In contrast, a "server system" is  
21 a computer system that provides services to another computer system.

22 Note that a computer system may use the services of another computer system and  
23 yet still provide services to other computer systems. Thus, a client system in one context  
24



1 may also be a server system in another context. Similarly, a server system in one context  
2 may also be a client system in another context.

3 Reference is also be made to a “network server” or “data server.” These types of  
4 servers fit inside the “client system”/ “server system” dichotomy as described above and  
5 may provide services to other computers and/or use services of other computers  
6 simultaneously.

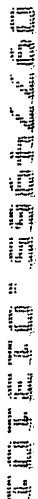
7 In this description and in the following claims the term HyperText Markup  
8 Language, or HTML refers to the authoring language used to create documents on the  
9 World Wide Web. HTML defines the structure and layout of a Web document by using a  
10 variety of tags and attributes. HTML is typically used to create documents that will be  
11 displayed on devices, such as a computer or computer system, using a web browser.

12 The term Uniform Resource Locator, or URL, refers to the global address of  
13 documents and other resources on the World Wide Web.

14 The term HyperText Transfer Protocol, or HTTP, refers to the underlying protocol  
15 used by the World Wide Web. HTTP defines how messages are formatted and transmitted,  
16 and what actions Web servers and browsers should take in response to various commands.  
17 For example, when you enter a URL in your browser, this actually sends an HTTP  
18 command “get” to the Web server directing it to return the requested Web page.

19 The term “displayable content” refers to content, which when received and output  
20 by a web browser such as Microsoft® Internet Explorer or Netscape Navigator®, has  
21 meaning to a human being. Displayable content primarily includes, but is not limited to,  
22 visual text or images or audio sounds, which when output have meaning to a human being.  
23 Displayable content may take the form of an HTML document.

24



1 In contrast, the term “non-displayable data” refers to data, which if sent to a web  
2 browser, such as Microsoft® Internet Explorer or Netscape Navigator®, in its native  
3 format would not be output in a manner that has meaning to a human being.

4 The term Dynamic Link Library, or DLL, refers to a library of executable functions  
5 or data that can be used by a Microsoft® Windows® application. Typically, a DLL  
6 provides one or more particular functions and a program accesses the functions by creating  
7 either a static or dynamic link to the DLL. A static link remains constant during program  
8 execution while a dynamic link is created by the program as needed. DLLs can also  
9 contain just data. A DLL can be used by several applications at the same time. Some DLLs  
10 are provided with the Microsoft® Windows® operating system and available for any  
11 Windows application. Other DLLs are written for a particular application and are loaded  
12 with the application.

13 While Figure 1 and the corresponding discussion above provide a general  
14 description of a suitable environment in which the invention may be implemented, it will  
15 be appreciated that the features of the present invention may be practiced in association  
16 with a variety of configurations. Therefore, the components illustrated in Figure 1 provide  
17 only one exemplary environment for practicing the present invention. It should be  
18 understood that the present invention, as described in connection with Figure 2, might also  
19 be practiced in a configuration that additionally includes one or more of the components as  
20 shown in Figure 1.

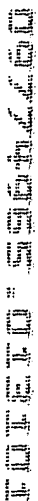
21 Figure 2 illustrates a network system 200 suitable for implementing the principles  
22 of the present invention. The network system 200 includes network devices 201, network  
23 server 202 and data server 203. Network devices 201 includes one or more network  
24

1 devices, including network device 201c. Network server 202 includes browser type  
2 determination module 212, language type determination module 222, template  
3 identification module 232 and rendering module 242. Data server 203 includes data store  
4 213. Although not required, each of the network devices 201, network server 202 and data  
5 server 203 may be structured as described above for computer 120. The components of the  
6 network system 200 mentioned above work together to implement the principles of the  
7 present invention.

8 Figure 3 illustrates a method 300 for rendering data from the data server 203 to  
9 create displayable content in accordance with the present invention. The method of Figure  
10 3 will be described in the context of the network system illustrated in Figure 2. Thus,  
11 elements of both Figures 2 and 3 will be referred to frequently in describing the operation  
12 of the present invention. The functionality of each of the components of Figure 2 will  
13 become clear from the following description relating to both Figure 2 and Figure 3.

14 In operation, network device 201c generates a request for displayable content. This  
15 request for displayable content may be, for example, an HTTP request making a request for  
16 an HTML document. The web browser 211c (e.g., one of the application programs 136 of  
17 Figure 1) associated with the network device 201c may generate the HTTP request which  
18 may be passed to the operating system (e.g., operating system 135 of Figure 1) to thereby  
19 transmit the HTTP request to the network server 202. The network server 202 then  
20 receives the request for the displayable content (act 301). This is illustrated by arrow 1 in  
21 Figure 2.

22 As an application example, suppose the user of the network device 201c is  
23 traveling with the network device 201c, but desires to view her electronic mail in-box to  
24 keep up-to-date. The user would then use the user interface of the web browser 211c to



1 indicate such a desire. In response, the web browser 211c would generate an HTTP  
2 request for the in-box content and use the operating system to transmit the HTTP request to  
3 the network server 202.

4 Note that network device 201c, network server 202 and data server 203 are  
5 illustrated in Figure 2 and described in the application example as being physically  
6 separate computer systems. However, network device 201c, network server 202 and data  
7 server 203 may also be partially or fully integrated with each other. For instance, if  
8 network device 201c and network server 202 were integrated, requests for displayable  
9 content would not be transmitted using network messaging. The web browser 211c of the  
10 network device 201c may pass the request to the network server 202 as a call through an  
11 Application Program Interface (API).

12 Upon receiving the request for the displayable content (act 301), network server  
13 202 identifies a template that corresponds to the requested content (act 302). In figure 2,  
14 network server 202 includes browser type determination module 212, language type  
15 determination module 222, and template identification module 232. Browser type  
16 determination module 212 determines the type of browser that the requested content will  
17 be sent to. The browser type may be explicitly identified in the request for content.  
18 However, the browser type may also be determined by other characteristics included in the  
19 request, such as the type of operating system that sent the request. Likewise, language  
20 type determination module 222, determines the language the requested displayable content  
21 will be sent in. This may also be explicitly identified in the request or may be determined  
22 from other characteristics included in the request.



1 Template identification module 232 uses the determinations made by browser type  
2 determination module 212 and language type determination module 222 in identifying a  
3 template that corresponds to the requested displayable content.

4 For example, in the practical example, a user transmits an HTTP request for her in-  
5 box to the network server. The browser 211c may automatically include an identifier  
6 identifying the version and type of the web browser as well as the preferred language of the  
7 user. The browser type determination module 212 and the language type determination  
8 module 222 may extract the express web browser type and language indications in the  
9 request and pass the results to the template identification module 232. The template  
10 identification module 232 then identifies the appropriate in-box template based on the web  
11 browser type and language.

12 Note that the various modules 212, 222, 232 and 242 are illustrated functionally as  
13 being separate modules. However, any two or more of these modules may be combined  
14 into a single integrated module. In addition, any single module may be broken down into  
15 multiple modules that perform more specific tasks.

16 Template identification module 232 may also use other data included in the request  
17 for content from network device 201c in making template identification or may use other  
18 already known data. Such data may include the bandwidth available to send the requested  
19 content, operating systems being used by computers attached to network system 200, or  
20 transient network conditions on network system 200. Template identification module 232  
21 may also use information about the item being bound to help determine the exact template  
22 to fetch. For example, an email you've received vs. an email you are composing – both  
23 email messages – result in different templates for doing their rendering.

24

1 Since such data may change over time, one feature of the template architecture is  
2 that templates may initially be coded to display data to different types of web browsers, in  
3 different languages or coded to account for other changes that may occur. That is, only  
4 one template need be created for a view and this template will be properly rendered on a  
5 variety of web browsers and languages. As a result, there is no need for extensive recoding  
6 to add functionality for a new browser type or language. For instance, referring to the  
7 application example of requesting an electronic mail in-box, an identified template may  
8 process requested displayable content in one way when the user of network device 201c  
9 requests access over a dial-up line, such as wide area network 152. However, the same  
10 template may process requested displayable content in different way when the user  
11 requests access over a permanent network connection, such as local area network 151.

12 Figure 4 shows example source code 400 in which custom tags are included in an  
13 HTML document. Such source code, when compiled, results in a template data structure  
14 that might be identified in act 302. Although the network server 202 does not read the  
15 source code of the template at run time (the compiled template data structure is instead  
16 used), the source code may be used to explain the features and functionality of the  
17 template.

18 At line 401, the browser type is checked to determine if the browser the requested  
19 content will be returned to is the default browser type. If the browser is the default type,  
20 then at the time data is rendered, lines 402 through 408 will be processed. If the browser is  
21 not the default type, then at the time data is rendered, lines 410 through 416 will be  
22 processed.

23 Lines 402-405 and 407-408 are simple HTML code, when these lines are processed  
24 they will not be changed. Line 406 includes a customized tag. The particular tag included

1 at line 406 is referred to as an xst-tag of which there are several types. The example given  
2 at line 406 is a call to get data, or "get-value," from "DAV:href," which may be a portion  
3 of the data stored in data store 213. The type attribute is used to specify how the data  
4 should be encoded at the time the template is processed. In this case "href" means that the  
5 data should be URL-encoded when the template is processed. The purpose of the  
6 customized tags is to provide instructions to the network server 202 identifying the non-  
7 displayable data, and the processing that is to be performed on the non-displayable data.  
8 The customized tags may be of any format so long as the network server 202 is able to  
9 interpret the customized tags as represented in the template. The processing of the non-  
10 displayable data includes processes to present the non-displayable data so that the data may  
11 be displayed on a Web browser.

12 In Figure 4, customized tags for the default Web browser and other types of  
13 browsers are similar. However, this is not required. It may be that different browser types  
14 have different display capabilities. This may require accessing differing sets of data from  
15 data store 213 in order to properly display the data. It may also be that certain functions  
16 need to be performed on the data before display on particular types of browsers. Also,  
17 different customized tags may be used depending on the language the requested content  
18 will be displayed in, other characteristics of the request, or network conditions on network  
19 system 200 at the time the request is made.

20 While Figure 4 depicts a template including two customized tags, any number of  
21 tags may actually be included. For instance, referring back the example of accessing an  
22 electronic mail in-box, when retrieving the contents of the in-box a template may include  
23 customized tags for the "To," "From," "Subject," "Date," and "Time" fields of each  
24 electronic mail message included in the in-box.

1 After a template is coded, it is compiled into a form that may be represented by a  
2 data structure. Figure 5 shows an exemplary data structure used to represent a template.  
3 Data structure 500 includes data dictionary 501, template constants 502, functions 503,  
4 token information table 504, and HTML data 505.

5 Data dictionary 501 identifies those data items that were represented in the template  
6 source code 400 and that are stored in data store 213. Template constants 502 includes  
7 constant data that will be used when rendering the template. For instance, template  
8 constants 502 may include references to things such as the words "To" and "From" if the  
9 template were being used to create a view to an electronic mailbox. Template constants  
10 502 may also be used to hold formats for things such as time and date. Functions 503  
11 references functions that are built into rendering module 242. Such functions may be used  
12 to manipulate data before it is sent as content to network device 201c. Token information  
13 table 504 correlates functions, references to data, and constants to the relative position in  
14 the HTML body. HTML data 505 is what is left out of a template after the template  
15 renderer removes all the custom tags. In figure 4, HTML data 505 would refer to lines  
16 402-405, 407, 408, 410-413, 415 and 416.

17 After a template is identified in act 302, network server 202 accesses data from data  
18 server 203, as illustrated by arrow 2 in Figure 2 (act 303). One computer system accessing  
19 data from another computer system may be performed by any number of methods well  
20 known in the art. As illustrated in Figure 2, network server 202 and data server 203 are two  
21 physically separate computer systems. However, network server 202 and data server 203  
22 may also be integrated so that the request for content need not be transmitted using  
23 network messaging. In this instance, modules on network server 202, such as template  
24

1 identification module 232, rendering module 242 or other modules, may pass the request to  
2 the data server 203 as a call through an Application Program Interface (API).

3 After data is accessed from data server 203, the network server 202 performs a step  
4 for constructing the requested displayable content (step 308). In one embodiment, step 308  
5 may include an act of including displayable portions of an identified template in the  
6 requested content (act 304). Displayable portions may include HTML directives that are  
7 included as part of a template or other displayable content.

8 Step 508 may also include an act of the network server processing non-displayable  
9 data accessed from a data server (act 305). Act 305 may include processing custom tags  
10 that are included in the HTML text, as illustrated in Figure 4. In one example, rendering  
11 module 242 parses a compiled template to detect customized tags, but does not modify any  
12 native HTML. Rendering module 242 generates a preliminary view (e.g., a web page) of  
13 the data in data store 213 that was accessed in act 303. Such data may be the contents of  
14 the user of network device 201c's electronic mail in-box. An advantage of this format is  
15 that since the data is processed at run time, that is, at time the template is processed, and  
16 templates are in their compiled format when processed, there is reduced risk of  
17 unauthorized access to data. To directly access the data in data store 213, a user would  
18 have to decompile a template.

19 Another design feature, which may be included in rendering module 242, allows  
20 reference to localized versions of things such as strings and date/time formats and then this  
21 data is rendered at run time based on the language that will be used to output the requested  
22 content. Localized data for a language is stored in files that are external to the code of the  
23 rendering engine and templates. In a Microsoft® Windows environment, these may take  
24 the form of resource DLL files. To add support for a new language, an administrator need

1 only install a resource DLL for an appropriate language. No modification to the rendering  
2 engine code or to any templates is required. Not only does this provide an easy way to  
3 quickly add support for additionally languages, but the rendering process is more efficient  
4 because it functions the same using any language.

5 This localization benefit may be obtained by including a generic representation of  
6 template constants in the template source code. For example, instead of directing the  
7 rendering of words such as "TO" or "FROM", a customized tag may direct the rendering  
8 of the localized versions of these fields. For example, the rendering module 242  
9 determines from the language type determination module 222, the precise language used  
10 (e.g., English). The network server 202 then identifies the DLL file corresponding to this  
11 language and extract the appropriate words to convey the intended meaning (e.g., "TO"  
12 and "FROM"). If the localized language were French or Mandarin, for example, different  
13 DLL files would be access to retrieve the relevant equivalent of "TO" and "FROM".

14 Step 308 may also include an act of the network server then including the  
15 processed non-displayable data, which is now displayable content as a result of the  
16 processing, in the requested content (act 306). After this act, both the displayable portion  
17 and the processed non-displayable data are included in a finalized view (e.g., a web page)  
18 to the data in data store 213 that was accessed in act 303

19 Network server 202 then sends the requested content to a network device (act 307).  
20 In Figure 2 the requested content is illustrated as being sent to network device 201c by  
21 arrow 3. However, the requested content may also be sent to another network device  
22 attached to network 200, or to other network devices on other networks attached to  
23 network system 200, such as the Internet.

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24

The present invention may be embodied in other specific forms without departing from its spirit or essential characteristics. The described embodiments are to be considered in all respects only as illustrative and not restrictive. The scope of the invention is, therefore, indicated by the appended claims rather than by the foregoing description. All changes, which come within the meaning and range of equivalency of the claims, are to be embraced within their scope.

What is claimed and desired secured by United States Letters Patent is: